surface_sense

December 1, 2021

We implement surface sensing by varying k_ret_on to create a delayed retraction after contacting the surface

[1]:
```
import os, sys
join = os.path.join
import numpy as np
import pandas as pd
import stats
import json
import matplotlib.pyplot as plt
from tabulate import tabulate
import SALib.analyze as sal
#
import rtw
import _fj
import txtdata
import sobol
import twanalyse
import twutils

from sobol import collect_obs, compute_sobol, format_sobol
```

```
WARNING: did not find local config.txt, default params loaded
```

[2]:
```
# start with sobol at
notedir, notename = os.path.split(os.getcwd())
simdir = "/home/dan/usb_twitching/run/b2392cf/cluster/sobol_ssense"
lookup = sobol.read_lookup(simdir)
problem = sobol.read_problem(simdir)
twutils.print_dict(problem)
```

```
{
        "num_vars": 6,
        "names": [
                "k_ext_off",
                "dwell_time",
                "pilivar",
                "anchor_angle_smoothing_fraction",
                "k_spawn",
                "k_ret_on"
        ],
        "bounds": [
                [
                        0.2,
                        1.0
                ],
                [
                        0.5,
                        3.0
                ],
                [
                        1.0,
                        20.0
                ],
                [
                        0.125,
                        1.0
                ],
                [
                        0.5,
                        5.0
                ],
                [
                        0.25,
                        2.5
                ]
        ]
}
```

[3]:

```python
# first job is to check that the simulation and analysis actually ran
stats_mask = np.array([os.path.exists(join(simdir, udir, "local.json")) for udir in lookup[0]])
num, denom = np.count_nonzero(stats_mask), len(lookup[0])
print('ran samples {}/{} ({:.1f}%)'.format(num, denom, 100*float(num)/denom))
```

```
ran samples 8192/8192 (100.0%)
```

[4]:

```python
# load summary statistics
objectives = ['lvel.mean', 'deviation.var', 'qhat.estimate', 'ahat.estimate',
    'nbound.mean', 'ntaut.mean']
Y, lduid = sobol.collect(objectives, targetdir=simdir, alldata=True)
```

[5]:

```python
# check for missing data
missing = sobol.check_missing(lookup, Y)
if missing:
    print(tabulate(missing, headers=["objective", "nan data"]))
```

```
else:
    print("No nan data")
```

```
No nan data
```

[6]:

```
import seaborn as sns

# sns.histplot(Y["lvel.mean"])
# sns.histplot(Y["deviation.var"])
# sns.histplot(Y["qhat.estimate"])
```

[7]:

```
second_order = False
Si = sobol.compute_sobol(problem, Y, second_order=second_order)
Si["lvel.mean"]["ST"]

dftable1, dftableT = sobol.format_sobol(problem, Si)
```

[8]:

```
from IPython.display import display, HTML
display(HTML(dftable1.to_html()))
display(HTML(dftableT.to_html()))

with open("/home/dan/usb_twitching/notes/sensitivity/tex/ssense_table.tex",␣
 ↪"w") as f:
    f.write(dftableT.to_latex())
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

We see that the retraction delay (k_ret_on) has almost no effect on the dynamics. Despite influencing nbound.mean it does not change ntaut.mean. We do not allow the extension motor to force pili into bent configurations but we do allow "bent" (l_ideal < l_contour) configurations due to attachment and changing body position. We expect pili which are not retracting to rarely be involvd in the dynamics This may not be the case if the bacteria has low or negative persistence e.g. In walking state. TODO redo this simulation in walking state